Notice of Allowability	Application No.	Applicant(s)	
	10/717,941	BLINICK ET AL.	
	Examiner	Art Unit	
	BEN C. WANG	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address--

All claims being allowable, PROSECUTION ON THE MERITS IS (OR REMAINS) CLOSED in this application. If not included herewith (or previously mailed), a Notice of Allowance (PTOL-85) or other appropriate communication will be mailed in due course. **THIS NOTICE OF ALLOWABILITY IS NOT A GRANT OF PATENT RIGHTS.** This application is subject to withdrawal from issue at the initiative of the Office or upon petition by the applicant. See 37 CFR 1.313 and MPEP 1308.

1. ☒ This communication is responsive to *the amendment dated November 12, 2009*.

2. ☒ The allowed claim(s) is/are *1-3, 9-16, 18, 20-23, and 28-36 (renumbered as 1-25)*.

3. ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
   a) ☐ All b) ☐ Some* c) ☐ None of the:
      1. ☐ Certified copies of the priority documents have been received.
      2. ☐ Certified copies of the priority documents have been received in Application No. _____ .
      3. ☐ Copies of the certified copies of the priority documents have been received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

   * Certified copies not received: _____.

Applicant has THREE MONTHS FROM THE "MAILING DATE" of this communication to file a reply complying with the requirements noted below. Failure to timely comply will result in ABANDONMENT of this application.
THIS THREE-MONTH PERIOD IS NOT EXTENDABLE.

4. ☐ A SUBSTITUTE OATH OR DECLARATION must be submitted. Note the attached EXAMINER'S AMENDMENT or NOTICE OF INFORMAL PATENT APPLICATION (PTO-152) which gives reason(s) why the oath or declaration is deficient.

5. ☐ CORRECTED DRAWINGS (as "replacement sheets") must be submitted.
   (a) ☐ including changes required by the Notice of Draftsperson's Patent Drawing Review (PTO-948) attached
      1) ☐ hereto or 2) ☐ to Paper No./Mail Date _____.
   (b) ☐ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No./Mail Date _____.

   Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the front (not the back) of each sheet. Replacement sheet(s) should be labeled as such in the header according to 37 CFR 1.121(d).

6. ☐ DEPOSIT OF and/or INFORMATION about the deposit of BIOLOGICAL MATERIAL must be submitted. Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

Attachment(s)

1. ☐ Notice of References Cited (PTO-892)
2. ☐ Notice of Draftperson's Patent Drawing Review (PTO-948)
3. ☐ Information Disclosure Statements (PTO/SB/08), Paper No./Mail Date _____
4. ☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material
5. ☐ Notice of Informal Patent Application
6. ☒ Interview Summary (PTO-413), Paper No./Mail Date *20100202* .
7. ☒ Examiner's Amendment/Comment
8. ☒ Examiner's Statement of Reasons for Allowance
9. ☐ Other _____.

/Ben C Wang/	/Michael J. Yigdall/
Examiner, Art Unit 2192	Primary Examiner, Art Unit 2192

DETAILED ACTION

1. Applicant's amendment dated November 12, 2009, responding to the Office action mailed August 12, 2009 provided in the rejection of claims 1-3, 9-16, 18, 20-23, and 28-36, wherein claims 1, 10, 13, 20, and 29-36 have been amended.

EXAMINER'S AMENDMENT

2. An examiner's amendment to the record appears below. Should the changes and/or additions be unacceptable to applicant, an amendment may be filed as provided by 37 CFR 1.312. To ensure consideration of such an amendment, it MUST be submitted no later than the payment of the issue fee.

3. Authorization for this examiner's amendment was given in a telephone interview with Mr. Kunzler (Reg. No. 38,527) on February 3, 2010 to further amend claims 10, 13, 20, and 29-36 (see Examiner's Amendment below) and thus to obviate any potential 35 U.S.C 112, second paragraph issues and to place the claims in the condition for allowance.

4. The application has been amended as follows:

IN THE CLAIMS,

Please amend claims 10, 13, 20, and 29-36 as follows:

1. (Previously Presented) An apparatus for updating an embedded code image in a host bus adapter performing high speed data transfer between a host system and a storage system, comprising a processor, a main memory, and a temporary memory, the main memory and the temporary memory storing Microcode Reconstruct and Boot (MRB) format code images, and the apparatus further comprising:

the processor executing executable code stored on the main memory occupied by and used by an old code image comprising a first MRB format header, the executable code comprising:

a loader stored in the main memory and loading a new code image comprising a second MRB format header into the temporary memory;

a branch module stored in the main memory causing the processor to execute a bootstrap module within the new code image;

the bootstrap module accessing an old code image version number pointed to by the first MRB format header and a new code image version number pointed to by the second MRB format header;

the bootstrap module identifying:

incompatibilities between the old code image and the new code image from the old code image version number and the new code image version number;

a difference in initialization requirements for storage registers, memory, and hardware devices; and

a difference in size and location between the old code image and the new code image;

the bootstrap module further:

accessing capability information for the old code image and capability information for the new code image;

identifying a difference between the capability information; and

reconciling the incompatibilities by

changing an initialization order;

converting a format of a data structure of the old code image to a format compatible with a data structure of the new code image; and

associating persistent data of the old code image with the new code image, such that

the persistent data is available in response to execution of a run-time segment of the new code image; and

a copy module copying the new code image into the main memory space occupied by the old code image.

2. (Previously Presented) The apparatus of claim 1, wherein the old code image is updated concurrently with normal execution of transactions by the apparatus.

3. (Previously Presented) The apparatus of claim 1, the executable code further comprising an initialization module initiating execution of the run-time segment.

4. (Canceled)

5. (Canceled)

6. (Canceled)

7. (Canceled)

8. (Canceled)

9. (Previously Presented) The apparatus of claim 31, wherein identifying incompatibilities further comprises identifying a difference between the format of the data structure used by the old code image and the format compatible with the data structure used by the new code image.

10. (Currently Amended) An apparatus for updating an embedded code image in a host bus adapter performing high speed data transfer between a host system and a storage system, comprising a processor, a main memory, and a temporary memory, the main memory and the temporary memory storing Microcode Reconstruct and Boot (MRB)~~MRB~~ format code images, the apparatus further comprising:

the processor executing executable code stored on the main memory occupied by and used by an old code image comprising a first MRB format header, the executable code comprising

a loader loading a new code image comprising a second MRB format header into the temporary memory;

a branch module stored in the main memory causing the processor to execute a bootstrap module within the new code image;

the bootstrap module accessing an old code image version number pointed to by the first MRB format header

and a new code image version number pointed to by the second MRB format header;

the bootstrap module identifying:

incompatibilities between the old code image and the new code image from the old code image version number and the new code image version number;

a difference in initialization requirements for storage registers, memory, and hardware devices; and

a difference in size and location between the old code image and the new code image;

the bootstrap module further:

accessing capability information for the old code image and capability information for the new code image;

identifying a difference between the capability information; and

reconciling the incompatibilities by changing an initialization order;

converting a data structure of the old code image to a format compatible with a data structure of the new code image prior to

copying the new code image into the memory space occupied by the old code image; ~~and~~

associating persistent data of the old code image with the new code image, such that the persistent data is available in response to execution of a run-time segment of the new code image;~~.~~ and

copying the new code image into the main memory space occupied by the old code image.

11. (Previously Presented) The apparatus of claim 10, the executable code further comprising a copy module copying the new code image over the old code image in the main memory after the incompatibilities have been reconciled.

12. (Previously Presented) The apparatus of claim 10, wherein identifying incompatibilities further comprises identifying a difference between the format of the data structure used by the old code image and the format compatible with the data structure used by the new code image.

13. (Currently Amended) A system for updating an embedded code image in a host bus adapter performing high speed data transfer between a host system and a storage system , the system comprising a processor, a main memory, and a temporary memory, the main memory and the temporary memory storing Microcode Reconstruct and Boot (MRB)~~MRB~~ format code images, the system further comprising:

the main memory storing an old code image comprising a first MRB format header;

the temporary memory separate from the main memory and storing a new code image comprising a second MRB format header;

a processor executing executable code of the old code image and the new code image, the executable code comprising;

a loader stored in the main memory and loading the new code image into the temporary memory;

a branch module stored in the main memory causing the processor to execute a bootstrap module within the new code image; and

the bootstrap module accessing an old code image version number pointed to by the first MRB format header and a new code image version number pointed to by the second MRB format header, the bootstrap module identifying:

incompatibilities between the old code image and the new code image from the old code image version number and the new code image version number;

a difference in initialization requirements for storage registers, memory, and hardware devices; and

a difference in size and location between the old code image and the new code image;

the bootstrap module further

accessing capability information for the old code image and capability information for the new code image;

identifying a difference between the capability information; and

reconciling the incompatibilities by

changing an initialization order;

converting a format of a data structure of the old code image to a format compatible with a data structure of the new code image; ~~and~~

associating persistent data of the old code image with the new code image, such that the persistent data is available in response to execution of a run-time segment of the new code image~~.~~; and

copying the new code image into the main memory space occupied by the old code image.

14. (Previously Presented) The system of claim 13, the executable code further comprising a copy module stored within the new code image overlaying the new code image in main memory with the old code image in response to reconciling the incompatibilities.

15. (Previously Presented) The system of claim 14, the loader loading the new code image into the temporary memory in response to an interrupt.

16. (Original) The system of claim 15, wherein the update module stores the old code image pointer and the new code image pointer in the data structure.

17. (Canceled)

18. (Previously Presented) The system of claim 33, wherein the bootstrap module further reconciles the incompatibilities by updating modules that interface with the new code image.

19. (Canceled)

20. (Currently Amended) A method for updating an embedded code image in a host bus adapter performing high speed data transfer between a host system and a storage system, comprising a processor, a main memory, and a temporary memory, the main memory and the temporary memory storing Microcode Reconstruct and Boot (MRB)~~MRB~~ format code images, the method further comprising:

loading, by use of the processor, a new code image comprising a second MRB format header into the temporary memory location separate from the main memory space occupied by and used by an old code image comprising a first MRB format header;

executing a bootstrap module within the new code image;

accessing an old code image version number pointed to by the first MRB format header and a new code image version number pointed to by the second MRB format header;

identifying, using the bootstrap module:

incompatibilities between the old code image and the new code image from the old code image version number and the new code image version number:

a difference in initialization requirements for storage registers, memory, and hardware devices; and

a difference in size and location between the old code image and the new code image;

accessing, using the bootstrap module, capability information for the old code image and capability information for the new code image;

identifying, using the bootstrap module: a difference between the capability information;

reconciling, using the bootstrap module executed by the processor, the incompatibilities by

changing an initialization order;

converting a data structure of the old code image to a format compatible with a data structure of the new code image using bootstrap code of the new code image; and

associating persistent data of the old code image with the new code image, such that the persistent data is available in response to execution of a run-time segment of the new code image; and

copying the new code image into the main memory space occupied by the old code image.

21. (Previously Presented) The method of claim 20, wherein the new code image is copied to the main memory concurrently with execution of regular computer operations.

22. (Previously Presented) The method of claim 20, further comprising initiating execution of the run-time segment of the new code image.

23. (Previously Presented) The method of claim 20, wherein the new code image is copied into the main memory after the incompatibilities are reconciled.

24. (Canceled)

25. (Canceled)

26. (Canceled)

27. (Canceled)

28. (Previously Presented) The method of claim 34, wherein identifying incompatibilities further comprises identifying a difference between the format of the data structure used by the old code image and the format compatible with the data structure used by the new code image.

29. (Currently Amended) An apparatus for updating an

embedded code image in a host bus adapter performing high speed data transfer between a host system and a storage system, the apparatus comprising a processor, a main memory, and a temporary memory, the main memory and the temporary memory storing Microcode Reconstruct and Boot (MRB)~~MRB~~ format code images and the apparatus further comprising:

the processor executing executable code stored on the main memory occupied by and used by an old code image comprising a first MRB format header, the executable code comprising:

means for loading a new code image comprising a second MRB format header into the temporary memory;

means for causing the processor to execute a bootstrap module within the new code image;

means within the bootstrap module for accessing an old code image version number pointed to by the first MRB format header and a new code image version number pointed to by the second MRB format header;

means within the bootstrap module for identifying:

incompatibilities between the old code image and the new code image from the old code image version number and the new code image version number;

a difference in initialization requirements for storage registers, memory, and hardware devices; and

a difference in size and location between the old code image and the new code image;

means with the bootstrap module for accessing capability information for the old code image and capability information for the new code image and identifying a difference between the capability information;

means within the bootstrap module for reconciling the incompatibilities by

changing an initialization order;

converting a format of a data structure of the old code image to a format compatible with a data structure of the new code image using bootstrap code of the new code image; and

associating persistent data of the old code image with the new code image, such that the persistent data is available in response to execution of a run-time segment of the new code image; and

means for copying the new code image into the memory space occupied by the old code image.

30. (Currently Amended) An article of manufacture comprising a program storage medium readable by a processor and embodying one or more instructions executable by a processor to perform a method for

updating an embedded code image in a host bus adapter performing high speed data transfer between a host system and a storage system by use of the processor, a main memory, and a temporary memory, the main memory and the temporary memory storing Microcode Reconstruct and Boot (MRB)~~MRB~~ format code images, the method comprising:

loading, by use of the processor, a new code image comprising a second MRB format header into the temporary memory location separate from a memory space occupied by and used by an old code image comprising a first MRB format header;

causing the processor to execute a bootstrap module within the new code image;

accessing, using the bootstrap module, an old code image version number pointed to by the first MRB format header and a new code image version number pointed to by the second MRB format header;

identifying, using the bootstrap module:

incompatibilities between the old code image and the new code image from the old code image version number and the new code image version number;

a difference in initialization requirements for storage registers, memory, and hardware devices; and

a difference in size and location between the old code image and the new code image;

accessing, by using the bootstrap module, capability information for

the old code image and capability information for the new code image and identifying a difference between the capability information;

reconciling, using the bootstrap module executed by the processor, the incompatibilities by

changing an initialization order;

converting a format of a data structure of the old code image to a format compatible with a data structure of the new code image using bootstrap code of the new code image; and

associating persistent data of the old code image with the new code image, such that the persistent data is available in response to execution of a run-time segment of the new code image; and

copying the new code image into the main memory space occupied by the old code image.

31. (Currently Amended) The apparatus of claim 1, wherein the loader configures the temporary memory so that the executable code is executed directly from the temporary memory, the executable code further comprising an update module stored in the main memory maintaining an old code image pointer, a new code image pointer, capability fields storing the capability information, the old code image version number, the new code image version number, the old code image pointer, the new code

image pointer, the capability fields, the old code image version number, and the new code image version number used by the bootstrap module, wherein the bootstrap module follows the old code image pointer to locate an old code image header and a version field within the old code image header and follows the new code image pointer to locate a new code image header and a version field within the new code image header, the old code image header and the new code image header are organized according to the MRB format, the bootstrap module reading the capability information from the old code image and the new code image and storing the capability information of the old code image and the new code image in the capability fields, the capability information comprising an indication that an EMULEX FLASH Random Access Memory (RAM)~~RAM~~ is provided, the persistent data comprising login tables, and wherein reconciling incompatibilities between the old code image and the new code image further comprises adjusting configuration settings and parameter lists.

32. (Currently Amended) The apparatus of claim 10, wherein the loader configures the temporary memory so that the executable code is executed directly from the temporary memory, the executable code further comprising an update module stored in the main memory maintaining an old code image pointer, a new code image pointer, capability fields storing the capability information, the old code image version number, the new

code image version number, the old code image pointer, the new code image pointer, the capability fields, the old code image version number, and the new code image version number used by the bootstrap module, wherein the bootstrap module follows the old code image pointer to locate an old code image header and a version field within the old code image header and follows the new code image pointer to locate a new code image header and a version field within the new code image header, the old code image header and the new code image header are organized according to the MRB format, the bootstrap module reading the capability information from the old code image and the new code image and storing the capability information of the old code image and the new code image in the capability fields, the capability information comprising an indication that an EMULEX FLASH Random Access Memory (RAM)~~RAM~~ is provided, the persistent data comprising login tables, and wherein reconciling incompatibilities between the old code image and the new code image further comprises adjusting configuration settings and parameter lists and associating persistent data of the old code image with the new code image.

33. (Currently Amended) The system of claim 13, wherein the loader configures the temporary memory so that the executable code is executed directly from the temporary memory, the executable code further comprising an update module stored in the main memory maintaining an

old code image pointer, a new code image pointer, capability fields storing the capability information, the old code image version number, the new code image version number, the old code image pointer, the new code image pointer, the capability fields, the old code image version number, and the new code image version number used by the bootstrap module, wherein the bootstrap module follows the old code image pointer to locate an old code image header and a version field within the old code image header and follows the new code image pointer to locate a new code image header and a version field within the new code image header, the old code image header and the new code image header are organized according to the MRB format, the bootstrap module reading the capability information from the old code image and the new code image and storing the capability information of the old code image and the new code image in the capability fields, the capability information comprising an indication that an EMULEX FLASH Random Access Memory (RAM)~~RAM~~ is provided, the persistent data comprising login tables, and wherein reconciling incompatibilities between the old code image and the new code image further comprises adjusting configuration settings and parameter lists.

34. (Currently Amended) The method of claim 20, the method further comprising configuring the temporary memory so that the executable code is executed directly from the temporary memory, maintaining an old code

image pointer, a new code image pointer, capability fields storing the capability information, the old code image version number, the new code image version number, the old code image pointer, the new code image pointer, the capability fields, the old code image version number, and the new code image version number used by the bootstrap module, wherein the bootstrap module follows the old code image pointer to locate an old code image header and a version field within the old code image header and follows the new code image pointer to locate a new code image header and a version field within the new code image header, the old code image header and the new code image header are organized according to the MRB format, the bootstrap module reading the capability information from the old code image and the new code image and storing the capability information of the old code image and the new code image in the capability fields, the capability information comprising an indication that an EMULEX FLASH Random Access Memory (RAM)~~RAM~~ is provided, the persistent data comprising login tables, wherein reconciling incompatibilities between the old code image and the new code image further comprises adjusting configuration settings and parameter lists.

35. (Currently Amended) The article of manufacture of claim 30, the method further comprising configuring the temporary memory so that the executable code is executed directly from the temporary memory, maintaining an old code image pointer, a new code image pointer,

capability fields storing the capability information, the old code image version number, the new code image version number, the old code image pointer, the new code image pointer, the capability fields, the old code image version number, and the new code image version number used by the bootstrap module, wherein the bootstrap module follows the old code image pointer to locate an old code image header and a version field within the old code image header and follows the new code image pointer to locate a new code image header and a version field within the new code image header, the old code image header and the new code image header are organized according to the MRB format, the bootstrap module reading the capability information from the old code image and the new code image and storing the capability information of the old code image and the new code image in the capability fields, the capability information comprising an indication that an EMULEX FLASH Random Access Memory (RAM)~~RAM~~ is provided, the persistent data comprising login tables, and wherein reconciling incompatibilities between the old code image and the new code image further comprises adjusting configuration settings and parameter lists and associating persistent data of the old code image with the new code image.

36. (Currently Amended) The apparatus of claim 29, wherein loading means configures the temporary memory so that the executable code is executed directly from the temporary memory, the executable code further

comprising maintaining means stored in the main memory maintaining an old code image pointer, a new code image pointer, capability fields storing the capability information, the old code image version number, the new code image version number, the old code image pointer, the new code image pointer, the capability fields, the old code image version number, and the new code image version number used by the bootstrap module, wherein the bootstrap module follows the old code image pointer to locate an old code image header and a version field within the old code image header and follows the new code image pointer to locate a new code image header and a version field within the new code image header, the old code image header and the new code image header are organized according to the MRB format, the bootstrap module reading the capability information from the old code image and the new code image and storing the capability information of the old code image and the new code image in the capability fields, the capability information comprising an indication that an EMULEX FLASH Random Access Memory (RAM)~~RAM~~ is provided, the persistent data comprising login tables, and wherein reconciling incompatibilities between the old code image and the new code image further comprises adjusting configuration settings and parameter lists.

- **END OF AMENDMENT –**

Allowable Subject Matter

5. Claims 1-3, 9-16, 18, 20-23, and 28-36 (renumbered as 1-25) are allowed.

6. The following is an examiner's statement of reasons for allowance:

The cited prior art taken alone or in combination fails to suggest "... updating an embedded code image ... a main memory, and a temporary memory ... storing Microcode Reconstruct and Boot (MRB) format code images ...

a loader stored in the main memory and loading a new code image comprising a second MRB format header into the temporary memory;

a branch module stored in the main memory causing the processor to execute a bootstrap module within the new code image;

the bootstrap module accessing an old code image version number pointed to by the first MRB format header and a new code image version number pointed to by the second MRB format header;

the bootstrap module identifying:

incompatibilities between the old code image and the new code image from the old code image

version number and the new code image version number;

a difference in initialization requirements for storage registers, memory, and hardware devices; and

a difference in size and location between the old code image and the new code image;

the bootstrap module further:

accessing capability information for the old code image and capability information for the new code image;

identifying a difference between the capability information; and

reconciling the incompatibilities by

changing an initialization order;

converting a format of a data structure of the old code image to a format compatible with a data structure of the new code image; and

associating persistent data of the old code image with the new code image, such that the persistent data is available in response to execution of a run-time segment of the new code image; and

a copy module copying the new code image into the main memory space occupied by the old code image.", as recited in independent claims 1 and similarly recited in independent claims 10, 13, 20, 29, and 30.

7. Claims (2, 3, 9, 31), (11. 12, 32), (14-16, 18, 33), (21-23, 28, 34), (36), and (35) are considered allowable by virtue of their dependence on allowable independent claims 1, 10, 13, 20, 29, and 30 respectively.

8. Any comments considered necessary by applicant must be submitted no later than the payment of the issue fee and, to avoid processing delays, should preferably accompany the issue fee. Such submissions should be clearly labeled "Comments on Statement of Reasons for Allowance."

Conclusion

9. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ben C. Wang whose telephone number is (571) 270-1240. The examiner can normally be reached on 8:00-5:30 (EST/EDT), Monday through Friday.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300. Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system.

Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Ben C Wang/
Examiner, Art Unit 2192

/Michael J. Yigdall/
Primary Examiner, Art Unit 2192